Integrating Agile Software Development and Enterprise Architecture Management

Sebastian Hanschke, Jan Ernsting, Herbert Kuchen
European Research Center for Information Systems (ERCIS)
University of Münster
Email: {sebastianhanschke,jan.ernsting,kuchen}@uni-muenster.de

Abstract

Practitioners consider the focus of Agile Software Development and Enterprise Architecture Management as divergent. Following this view, two questions arise and are answered by this paper: whether and how agile methods such as Scrum can be used to create architecture deliverables and how enterprise architects can collaborate with agile software development teams. Based on expert interviews in a major German consultancy, a railway company, and an automotive OEM an integration of the TOGAF ADM and Scrum has been developed and evaluated following the Design Science research process. Both questions are answered by two different procedure models, one to be used on TOGAF's Enterprise Strategic and Segment Architecture Level and another for the Capability Architecture Level. The first enables the creation of the EA in form of Scrum projects. The second focuses on the necessary collaboration of empowered implementation teams with a central EA function.

1. Introduction

Although professionals are already trying to combine both Agile Software Development (ASD) and Enterprise Architecture Management (EAM), this combination has to the authors' knowledge been barely researched so far (cf. [1, p. 324]). According to Breivold et al. “it is hard to conclude neither that agile and architecture is like oil and water, nor that the two are the perfect marriage” [2, p. 36]). As such, EAM is not assumed by ASD and vice versa [3, p. 56f.]. Yet, the two “can be combined in a fruitful manner” [4, p. 3].

Integration approaches typically answer either of two questions: Whether and how can agile methods such as Scrum [5] be used to create architecture deliverables? How can enterprise architects collaborate with agile software development teams? Highlighting the importance of this collaboration, our paper focuses on both and contributes an integration of The Open Group Architecture Framework (TOGAF) [6] as an example of a EAM framework and Scrum, representing an ASD approach, to fill this gap. Due to their wide dispersion and thus relevance in practice [7, p. 2], TOGAF and Scrum have been chosen. Overall, the proposed integration offers an unprecedented level of detail and deals with the divergent focuses of ASD and EAM, e.g. short vs. long-term goals, bottom-up vs. top-down perspectives, single system vs. system landscape, etc. (cf. [8, p. 68]).

To answer the above questions, two research methods are combined. This paper follows the Design Science (DS) research process [9]. The knowledge required in some of the steps of the DS process was gathered by means of semi-structured interviews, which allow a rather flexible course of discussion. These interviews are characterised by openly formulated questions that serve as an orientation, to which interviewees can answer rather freely [10, p. 156]. They were conducted with EA and software development experts of a consultancy, a railway company, and an automotive OEM. Thus, problems that the integration has to overcome can be identified. In the subsequent phase, identified problems are used to derive objectives of a solution, based on which a concrete solution is developed. This solution was iteratively improved based on the Design and Development phase interviews. As the introduction of an integrated method would require a large investment in terms of money and time, and as effects are rather hard to measure, the Demonstration and Evaluation phases are replaced by several in-depth discussions of the final integration proposal with the experts.

First, we describe basic concepts of EAM and ASD. In Section 3 the integration is developed following the DS process. Next, we provide an overview of existing approaches. Finally, the conclusion summarises our findings and discusses future work.

1530-1605/15 $31.00 © 2015 IEEE
DOI 10.1109/HICSS.2015.492

Table 1. Phases of the TOGAF ADM [6, pp. 10f.]

Phase	Description
Preliminary Phase	Preparation and initiation activities necessary to setup the Architecture Capability including customisation of TOGAF and definition of Architecture Principles, usually only carried out once
A: Architecture Vision	Initial phase of an architecture development cycle: Definition of scope, identification of stakeholders, creation of the Architecture Vision, obtainment of stakeholder approval to proceed
B: Business Architecture	Development of Business Architecture to support the agreed Architecture Vision
C: Information Systems Architecture	Development of Information Systems Architecture to support the agreed Architecture Vision (i.e. Data and Application Architectures)
D: Technology Architecture	Development of Technology Architecture to support the agreed Architecture Vision
E: Opportunities & Solutions	Initial implementation planning and identification of ways to deliver the architecture defined in the previous phases
F: Migration Planning	Detailed Implementation and Migration Plan to move from Baseline to Target Architectures (Transition Architectures)
G: Implementation Governance	Architectural oversight of the implementation, governance function
H: Architecture Change Mgmt.	Procedures for managing change to the architecture
Requirements Mgmt.	Continuous process of managing architecture requirements throughout the ADM, influences activities/results of each of the eight main phases

2. Background

2.1. Enterprise Architecture Management

Although there is no agreement on a definition of the term Enterprise Architecture (EA) [11, p. 113], the definition by Lankhorst is widely-used and appropriate within the context of this paper. According to [12, p. 11], EA can be defined as “a coherent whole of principles, methods, and models that are used in the design and realization of an enterprises organizational structure, business processes, information systems, and infrastructure”. EAM is a holistic approach to continuously control and plan the development and the implementation of a company’s EA which seeks to ensure that the company’s goals and strategy are reflected in both business and IT. EAM enables operational, tactical, and strategic planning and does, consequently, not only look at the current As-Is architecture, but also gathers requirements of the different stakeholders with regard to the To-Be architecture. Despite the modelling of the EA, EAM also covers the management of the application portfolio and the IT infrastructure, as well as the monitoring of the project portfolio and each of its projects.

EA frameworks such as TOGAF or the Zachman Framework aim to ease the otherwise rather challenging execution of EAM [13, p. 205]. For example, the frameworks help with the design and establishment of EAM processes, e.g. to enable the effective and efficient creation of architecture models at an adequate level of detail [14, p. 65]. TOGAF itself consist of seven main parts, the Introduction, Architecture Development Method (ADM), ADM Guidelines and Techniques, Architecture Content Framework (ACF), Enterprise Continuum (EC), Architecture Reference Models, and the Architecture Capability Framework. Although all of these parts are meant to work together, it is, due to this modular structure, possible to select only parts, e.g. to incrementally adopt TOGAF or to merge parts of TOGAF into self-developed frameworks [6, p. 5]. It is thus possible that the integration described in Section 3 of this paper is limited to the ADM.

The ADM is the core of TOGAF. It is a holistic, step-by-step approach to develop an EA on different levels of detail, i.e. Enterprise Strategic, Segment, and Capability Architectures. The ADM consists of eight plus two phases, which are, apart from the usually non-recurring Preliminary and the continuously executed Requirements Management phase, enumerated from A to H (see Table 1). Although the graphical description of the ADM (see upper right of Figure 2) gives the impression that it is sequential, the ADM is iterative over all, within, and between phases. The ADM is meant to be tailored; steps can be adapted in scope or level of detail [6, p. 10].

2.2. Agile Software Development

In recent years, software development endeavours increasingly employ ASD. Its iterative and incremental nature [15, p. 5302] leads to short feedback loops resulting in improved stakeholder collaboration and the ability to deal with changing requirements [16, p. 340].

Scrum is one prominent ASD approach and provides a well-defined framework [5], [17], which can also be used in domains such as sales [18]. Through the interaction of roles, events, and artefacts in the Scrum life-cycle, a Product Increment is delivered at the end of each Sprint (see lower half of Figure 2). Each sprint lasts for a fixed time period of one to four weeks. During this time period, the team works on a number of tasks that it has committed to and is not disrupted from the outside. Not only the sprint, but most activities are time-boxed, i.e. they are strictly limited to a certain timespan. Scrum focuses on the use of feedback and retrospectives to enable continuous improvement. As Scrum's roots lie within the empirical process control and engineering, the process of how the software is developed during a sprint is not described. Scrum rather emphasises the cooperation of cross-functional individuals in small, self-organised teams.

Each member of a Scrum team has one of three roles, namely Product Owner, Scrum Master, or being a Development Team Member. The Product Owner (PO) is held responsible for the product at hand and is the main collaborator for stakeholders. He continuously maintains the Product Backlog (PBL) – a list of the stakeholders' requirements with regard to the product – and prioritises its items with respect to their expected return on investment. Furthermore, the PO accepts or rejects product increments during the Sprint Review. The Scrum Master (SM) serves the team and tries to ensure that they have understood and adhere to their process. Thus, he may help the PO in creating succinct PBL Items or remove impediments to the team. Providing the opportunity for introspection, SMs typically facilitate the Sprint Retrospective during which the Scrum team identifies and plans improvements for the next sprints. Development Team Members are responsible for creating the Product Increment. It is defined by the PBL Items that were selected as the Sprint Backlog during Sprint Planning. Facilitated by the SM, team members meet to synchronise their progress and identify impediments in the Daily Scrum.

3. The Integration of TOGAF and Scrum

The integration described in the following is based on semi-structured interviews [10, p. 156f.] conducted with experts of a consultancy, a railway company, and an automotive OEM. Initial ideas for the integration were discussed in a focus group [10, p. 195f.], before they were presented to the first interviewee. Each of the subsequent interviews has been slightly different, as an updated version of the integration was discussed, while the pretested interview guidelines remained unchanged. Two of the five focus group experts also participated in one of the ten individual interviews. Both the focus group and the interviews lasted one up to one and a half hours. According to Flick [10, p. 294f.] these were recorded to be transcribed and paraphrased by the interviewer. Next, the paraphrases were reduced and summarised to generate inputs for the integration design. Here the final version is presented.

3.1. Problem Identification

First of all, the different focuses of the two methods hinder their integration, while at the same time showing that they could clearly complement each other. EAM, on the one side, takes a rather top-down perspective directly derived from and even influencing business goals and strategy. It is used to set long-term goals and define a roadmap for reaching them. The focus of EAM is on long-term value. ASD is more likely used on a project level. It takes a bottom-up perspective, breaks down requirements into atomic entities, and implements them in short time-boxes. These short-term planning horizons enable the dynamic consideration of changing requirements and reduce the time-to-market. Thereby, the focus of ASD is on short-term value.

Using Scrum, while keeping an eye on the strategy and the goals of the overall enterprise, is not easy. Often teams lose sight of these objectives and try to achieve a project-specific rather than an enterprise-wide optimum. This problem may be reinforced by the self-empowerment. Especially when being used in complex settings, even ASD does need guidance as it can be provided by EA(M). Results have to be documented in detail, e.g. due to regulations or because teams are being replaced or contracted from external service providers; yet, ASD does not remove documentation altogether. Furthermore, due to the fact that Scrum is often used in pilot projects to develop new products, migration planning is often not considered. Here, Scrum might benefit from being enhanced by aspects from TOGAF. While agile approaches might control or steer too little and require more goal-orientation, EAM on the other hand can benefit from becoming more dynamic and develop a stronger focus on collaboration.

EA is often perceived to be created in an 'ivory tower'. There might be truth in this, as substantial effort is typically required before first results are delivered, for which the architects withdraw themselves. It does, however, create a distance between them and project teams. Thus, mistakes are identified relatively late and implementation teams often have to await architecture decisions. Architects are perceived to prevent teams from doing what they think is best for their project instead of enabling them to deliver the best possible solution. In the eyes of agile software developers, EAM forces to create too much unneeded documentation. Although TOGAF and other EA frameworks do not necessarily have to be as static and sequential as they are in many companies, most of the big transformation projects relying on them are not perceived to be agile. Furthermore, EAM and project (portfolio) management are not as tightly linked as needed. It remains to be seen how to derive projects from the To-Be IT landscape and how to ensure architecture compliance of these implementation projects. Thus, EA(M) might also benefit from the integration with Scrum.

3.2. Solution Objectives

In general, the solution should enable architects and software developers alike, to overcome the identified problems. At least one of the two methods will have to be adapted. According to the interviewees, TOGAF is well suited for this, as it is intended to be tailored. Moreover, premature changes to Scrum are said to impair its overall effect [19].

On the one hand, the integration should concentrate on the creation of the EA and its implementation in agile projects. Thereby, the first objective of the integration is to combine the different focuses of the two methods. The integration should enable the definition of long-term goals for and the concerted development of the IT landscape, while keeping an eye on the business strategy and the overall goals of the enterprise. At the same time it should facilitate adapting the EA to changing requirements of its stakeholders. On the other hand, the method should support the identification of reasonable projects to implement the architecture using Scrum.

From an enterprise architect's point of view, the integrated method is the answer to two questions. First, how to collaborate with agile software development teams. Second, taking into consideration the advantages of ASD, an architect might ask himself whether it can also be applied in his domain. Thereby, the solution should not be restricted to the Technical Architecture, but should be suitable to create the overall EA. In short, the complete TOGAF ADM should be considered by the solution. Architecture changes or the current status of the architecture should regularly be communicated. The solution should not require spending too much time on documenting the As-Is architecture, but should enable to start implementing architecture changes as quickly as possible. Instead of being driven by projects, the EA should trigger projects itself. Yet, this should not mean that projects are handed over a ready-made architecture for implementation. The EA is responsible for setting the context of implementation projects. As the EA

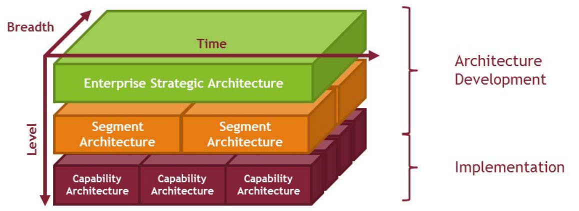

Figure 1. Classification Model for Architecture Landscape [6, p. 196]

triggers projects which might have interdependencies, an integrated method has to take care of the project (portfolio) management and migration planning. During the execution of the implementation projects, there should be close collaboration between the architects and the agile teams. As every architecture deliverable, which these projects create, is integrated into the overall EA, an up-to-date overview is created incrementally.

From the point of view of an agile software development team, the architecture should not be created by architects sitting on an 'ivory tower'. The integration should rather leverage the knowledge and expertise of the members of the development teams and leave as many of the decisions as possible to them. The teams should therefore be empowered to define solutions on their own, which have to consider limitations set by the current and future IT landscape. Apart from enabling teams to make their own decisions, interaction with the EA should be enabled. The integration should help to cross the chasm between the EA and the project teams and increase the collaboration between them.

3.3. Solution Design & Development

As an integrated approach fulfilling these objectives could not be found and does not exist according to the knowledge of the interviewed experts, it is developed below. Thereby, different levels of granularity on which architectures are created are distinguished. This paper follows the distinction proposed by TOGAF, assuming that architectures are created on three different levels (see Figure 1). Nevertheless, it assumes that the same procedure model can be used for the Enterprise Strategic and the Segment Architecture level.

The goal of the solution is to enable agility on all of these levels. On the Enterprise Strategic and the Segment Architecture Level, the proposed procedure model does thereby have a two-folded focus. On the one hand, it enables the agile creation of architecture models and other deliverables. On the other hand, it empowers the EA function to work together with the projects implementing this strategy, whenever they have a need for architecture input. Thereby the architecture compliance of decisions taken by the teams is ensured. On these levels, the TOGAF ADM is carried out as a number of Scrum projects. These Scrum projects set the framework based on which a detailed overall architecture can gradually be developed.

On the Capability Architecture Level, Scrum projects are used to implement architecture changes. As all interviewees agreed, that Scrum should only be changed if absolutely necessary, only small adaptations have been carried out. Thus, on this level, pure Scrum with connection points to the EA(M) is used. Despite a close collaboration with the EAM, it is also important to closely work together with all stakeholders. The implementation teams should, however, be empowered to take a large part of the decisions themselves. In doing so, the teams have to respect limiting factors set by the overall EA, but still, programmers should be seen as experts and not as assembly-line workers. The results of their decisions should directly be incorporated into the overall architecture. Future projects might benefit from the fact that they can get a complete and up-to-date overview of the system landscape among other possible merits.

3.3.1. Integration on the Enterprise Strategic and the Segment Architecture Level.

On the Enterprise Strategic or Segment Architecture Level a framework architecture is created and its refinement and implementation in vertical cuts through all architecture layers, as proposed by Scrum, are enabled. Therefore, the TOGAF ADM is split up into four different Scrum projects.

The Architecture Development Team (ADT) is the first of these four teams. It consists of an Architecture Product Owner (APO), an Architecture Scrum Master, and further Business, Information Systems, and Technology Architects as team members. As the name suggests, the ADT is responsible for developing the EA. It does thereby take care of the phases A to D and the Requirements Management phase of the TOGAF ADM (see Figure 2).

The Architecture Vision, which can be compared to the Product Vision of Scrum, is the starting point of the architecture development. It is developed by the APO along with the business management and directly represents the business strategy. As in Scrum, the Architecture Product Owner is commissioned with the development of the architecture according to this vision by the management. He starts to derive requirements, on the one hand from the Architecture Vision, and, on the other hand, by talking to stakeholders, gathering their expectations and needs with respect to the architecture. For the Requirements Management an Architecture Product Backlog (APBL) and User Stories are used, which are prioritised by the APO. The team does then select a certain amount of the highest-rated stories to work on during the sprint. Depending on the selected Architecture Product Backlog Item, Business Architecture, Information Systems, and/or Technology Architecture deliverables are created.

As the Implementation Teams are empowered Scrum teams, it is important not to pre-specify too much of the Capability Architecture on this level. Especially with respect to Information Systems and Technology Architecture, decisions on the upper levels should only be of strategic nature, e.g. with respect to standards or requiring long-term investments. Concrete, technical decisions are however taken by the technical experts in the Implementation Teams. As every architectural decision and every architecture deliverable created by these teams is sent back to the EA and incorporated into the overall architecture, a complete and up-to-date picture of the EA is created incrementally.

Especially when there is more than one ADT, the second type of Scrum teams, the Portfolio Management Team (PMT), becomes relevant. It fulfils the objectives of the Opportunities and Solutions and the Migration Planning phase of the TOGAF ADM (see Figure 2). The team is thereby responsible for keeping an overview of the various changes to the architecture that are planned by the different ADTs, resolving potential conflicts, and combining changes into projects. This separate team of portfolio management experts performs its tasks parallel to the architecture work (see Subsubsection 3.3.3). To get an idea of what the different ADTs are working on, the PO of the PMT meets with all APO in the Team of Product Owners (see Figure 2).

The PMT does then create an initial version of the Product Vision and the Product Backlog for each Implementation Team. These PBL contain architecture requirements that the team has to fulfil, e.g. interfaces that have to be implemented as well as items roughly describing the functionality to be implemented. Whenever such a PBL is ready for implementation, e.g. whenever the value of items is big enough to justify the commissioning of an Implementation Team, a new implementation project is triggered.

With the Implementation Teams being the third, the Architecture Change Management Team is the fourth type of Scrum teams. This separate Scrum team looking for trends, new technologies, or business models that require Architecture Change is however outside of the focus of this paper, which is on the development and implementation of the architecture.

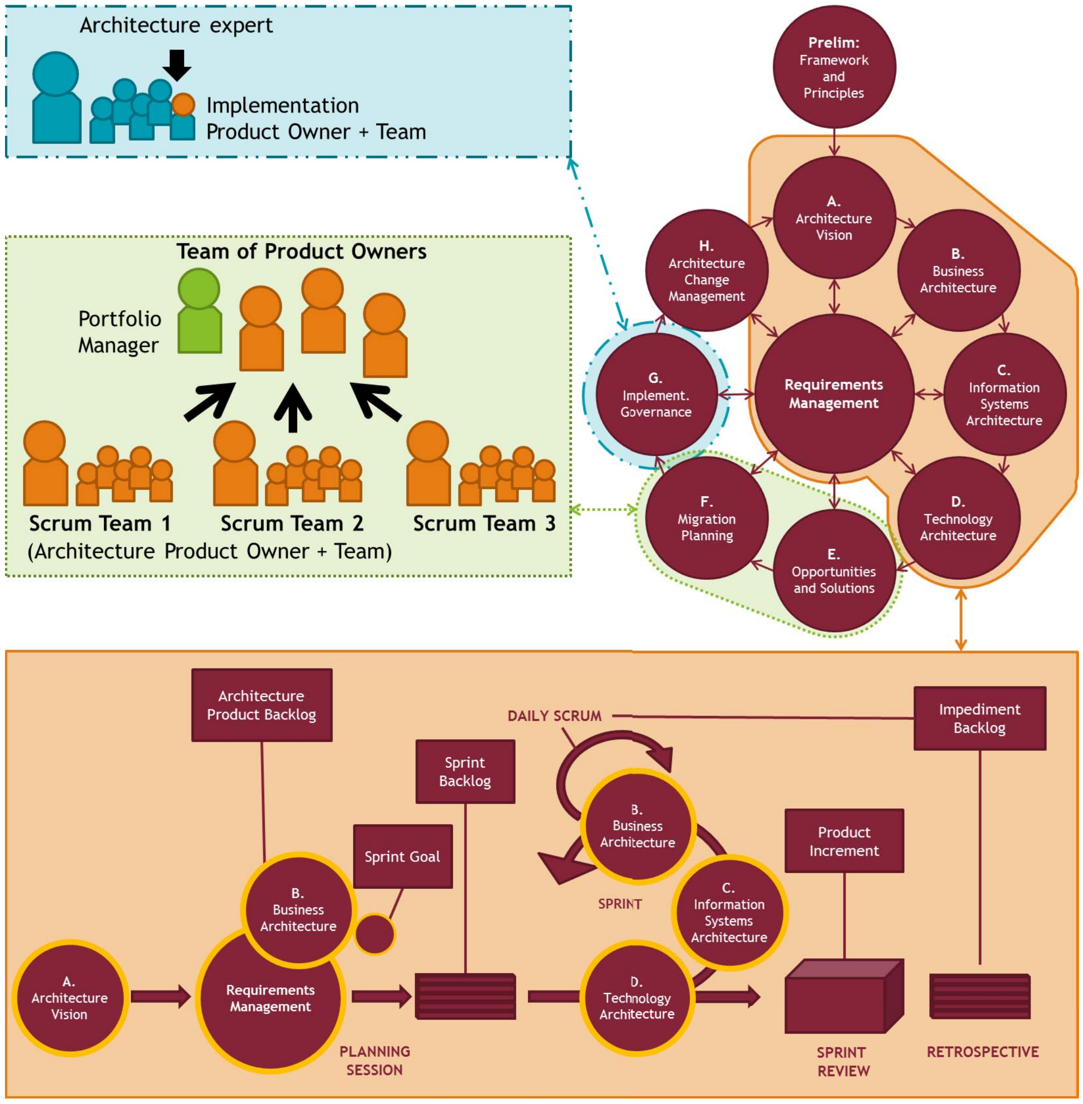

Figure 2. Overview of the Integration

3.3.2. Integration on the Capability Architecture Level. During the Implementation Governance phase the Implementation Team (ImplT) starts to develop the product according to the PBL and the Product Vision assigned to it. On this level, Scrum, only extended by interaction points with the EA, is used. During the implementation project three situations might occur:

- Architecture Deliverables might be required or created by the Implementation Team,
- Architecture Compliance should be checked, or
- Impediments might be caused by the EA function.

Whenever an Architecture Deliverable is required or created, the team should think about the nature of the deliverable. If a decision is product-specific, i.e. it does not have an effect on systems outside of the project's scope, and the team wants to use a standard technology, the empowered ImplT takes the decision itself. Every architecture deliverable created to document this decision is sent to the EA function to be included into the overall architecture. Whenever a team feels that a decision might have an impact on the overall EA or whenever the team requires further information about the system landscape it is obliged to contact the EA. Whenever the EA function is contacted and asked for architectural input, they create an Architecture Product Backlog Item. As the APO knows, that the ImplT needs this input to move on, he usually assigns a high priority to this item. In order to be able to use the architectural input provided, but also create documentation that can be incorporated into the overall architecture and can be re-used by other teams, architectural knowledge in the Scrum team is essential. Therefore, each team should at least have one Architecture Expert.

This demand or the creation of architecture deliverables can occur at different points of time in the Scrum lifecycle. Whenever a Product Backlog Item is selected or worked on, the team might have the need for clarification with regard to the EA. Furthermore, it does in some cases make sense to have the architects prepare a User Story from an architectural point of view before it is selected. Otherwise, impediments might result from the fact that teams have to wait for architecture decisions or else the number of architects has to be very high to ensure a short response time. In this case, the PBL Item might however still be discarded, so that the architects would perform unnecessary work. This preparation should therefore only be carried out for PBL Items with a very high complexity and a high likelihood of realisation in one of the next sprints.

Each of the ImplT's decisions is documented in form of architecture deliverables which are sent to the ADT to be incorporated into the overall EA. The Architecture Compliance of these decisions should however be checked before. This does already happen during the sprint, typically supported by advanced tool-support. Decisions of the team might then be automatically reported to the responsible architects, could directly be checked, and – if necessary – an alternative might be suggested. Furthermore, an architect might take part in the Sprint Review and confirm the architecture compliance to the PO, the customer, and other stakeholders.

The intense collaboration between ADTs and ImplTs can lead to Impediments. If an ADT is under high workload when an architecture deliverable is requested, it might be unable to directly provide the necessary information. This is an impediment, as the team has to wait due to an external factor. As Scrum suggests, it is the responsibility of the Scrum Master of the ImplT to make sure that this impediment is cleared away. Therefore, he might, for example, make sure, that a higher priority is assigned to this APBL Item, see whether the information can also be provided by another Architecture Team, etc.

3.3.3. Pipelining. Apart from the ad hoc collaboration, which has been described above, it is important to understand how sprints of the different teams are interconnected. Therefore, the concept of pipelining is used. The Architecture Development Team and the Portfolio Management Team start one sprint ahead of the Implementation Team (depending on the level of detail at which the EA is already documented more sprints might be required). Although the PMT does of course depend on the concrete design decisions taken by the ADT, they should be able to do their work in parallel. The results of these two teams are then handed to a number of ImplTs. While these teams work on the implementation, the ADTs and the PMT can already prepare the next architecture changes to be implemented.

As with pure software development teams working together using pipelining, problems arise when an ImplT does not manage to finish its work on time, e.g. because of a problem with the architecture that the ADT has designed. In this situation, rework or rectifications are required. As the Sprint Backlog of the ADT can however not be changed, a new APBL Item has to wait for the next sprint. One solution for this is a very short sprint length of the ADT.

This pipelining is supported by personal cooperation, meaning that members of each team are encouraged to regularly take part in the Daily Scrum of other teams they work with. Furthermore, an Architecture Community is established, to stimulate the exchange between people dealing with the EA.

3.4. Demonstration & Evaluation

The demonstration of the above integration would take several years. Therefore, demonstration could not be completely carried out, yet. Instead, the solution was presented to the interviewees as well as in several presentations and discussions.

Overall, the interviewees considered it promising to apply Scrum to areas other than software development and are looking forward to see, whether the integration, and especially the collaboration between EA and ASD, is going to work in practice. On the one hand, the EA (among other business functions) can thereby be closer aligned with agile projects. On the other hand, the EA function itself strives to become more agile, i.e. flexible and may be quicker to react. This last prospect of the EA becoming less cumbersome and creating results more quickly was neglected by some of the experts. They did however agree that single steps of the TOGAF ADM could benefit from increased agility. The integration would lead to a better prioritisation of the architecture deliverables to be created and would thus enable a directed, more purposeful EA. The idea of identifying an analogue of the product in software development in the EA and thus the integration presented in the previous chapter was seen as "the right way" to do so. According to the experts, it makes sense to have the EA teams trigger implementation projects.

Although the final version of the integration was well received by all interview partners, the experts did also point out potential problems. Handing down predetermined PBLs to Implementation Teams can be problematic. It remains to be seen, whether POs of these teams automatically prioritise these predetermined PBL Items in a way that ensures their implementation. Furthermore, as described above, the combination of Scrum's time-boxes and the concept of pipelining was considered to be a potential issue. Instead of working with short sprint-lengths, other (agile) methods, e.g. Portfolio Kanban [20] might therefore be better suited to be used on the more strategic layer. Approaches such as S-BPM [21] might help to render the phases B-D even more agile. Apart from these problems, there are a number of limitations and possible extensions, which should be overcome or at least thought of before the integration is introduced. From the authors' point of view considering the rest of TOGAF is the most interesting of these extensions. So far, the solution focuses on the active design of the EA. Furthermore, tools to support the integration should be developed or existing tools adapted.

4. Existing Approaches

When differentiating the above integration from existing approaches, it is worthwhile to consider the problems that such an integration has to overcome (cf. Subsection 3.1). Preparatory discussions with experts who have a background in EAM, ASD, or both hinted towards few existing integration approaches. Our literature research confirmed the experts' propositions. Breivold et al. [2] provide an overview of the few scientific sources. Similar to Buckl et al. [1], we do in the following consider non-scientific sources such as white papers or blog discussions due to their valuable contributions.

Instead of describing actual approaches, the majority of papers dealing with ASD and (Enterprise) Architecture focus on discussing their differences or similarities and the resulting challenges for an integration [22]–[24]. Buckl et al. [1] discuss the agility of current approaches. According to Yaghoubi and Babanezhad [25, p. 36], approaches either focus on the use of architecture in agile methods or try to make the creation of architecture work more agile. Consequently, approaches can be distinguished in those based on ASD (i.e. Scrum or another ASD method are extended by aspects of EAM) and approaches based on EA(M) (i.e. trying to enable a higher agility when using an EA(M) framework). In the following, a brief overview is given and notable approaches are presented.

4.1. Approaches based on ASD

Typically, ASD approaches such as Scrum, Extreme Programming [26], etc. have to be extended to enable the consideration of architecture [27], [3, p. 57]. These approaches are usually limited to software architecture, i.e. the architecture of the product being built. They do not consider the full software development life-cycle (SDLC) and often neglect the risk or consequences of failing to deliver large enterprise projects [28, pp. XVII & XXI].

Examples of more advanced approaches are the pattern based process for agile architectural modelling [29] and the proposal to use an Architecture Abstract Specification [30].

Open Unified Process (OpenUP), the Dynamic System Development Method (DSDM), and Disciplined Agile Delivery (DAD) [31, p. 10f.] are samples that take EA one step further and do so by combining practices from several core agile methods. With regard to EA, DAD already includes some of the required elements, e.g. the Architecture Owner role [28, p. 80], which can be thought of as a PO of the architecture (cf. Subsubsection 3.3.1). Although DAD can, according to its developers, easily be tailored to fit to the overall EA(M), no details on how to do this could be found.

Ambler and Lines [28, p. 1] advise that advanced approaches should no longer limit themselves to small, co-located teams. Dealing with larger teams or otherwise complicated settings, e.g. due to geographical distribution, regulation, complexity of the project, outsourcing, or legacy systems, should rather be the norm. They "recognize a basic need in enterprises for a level of rigor that core agile methods dismiss as not required such as governance, architectural planning, and modelling" [28, p. 2]. Ambler [32] highlights that EA should be business-driven, evolutionary, collaborative, focused on producing valuable artefacts, and an explicit part of overall delivery process.

4.2. Approaches based on EAM

Buckl et al. [1] and Edwards [33] investigate Scrum's applicability to EAM or the efficient management of the activities of an EA practice. Apart from these, every publication that we found focuses on TOGAF.

Edwards [34] identifies problems of TOGAF, e.g. that the ADM is perceived to be a serial process, and argues how these problems can be addressed on a conceptual level, without however proposing a usable solution. Alali [35] describes a mapping between TOGAF and ASD, which is however limited to the phases G and H of the ADM and thus focuses on governance metrics.

He does however hint at the different cycle times of TOGAF and Scrum as a potential problem. Ismail [36] proposes to use Scrum in Phases E to G, i.e. preparing and executing the implementation work, but also acknowledges a requirements gathering mismatch.

Apart from these rather general concepts or integration approaches which are limited to a subset of TOGAF phases, we found two approaches that consider the complete TOGAF ADM. The Benefits Led Enterprise Architecture Method (BLEAM) [37] is an agile EA framework based on TOGAF. At closer inspection though, the authors do only describe a TOGAF tailoring.

Bombosch [38] stresses the importance of culture for an integration of architecture and agile software development, e.g. he considers communication and the wisdom of crowds to be crucial. To him architecture should create a framework within which agile implementation teams navigate rather than providing specific programming instructions. Using TOGAF and Scrum he sketches an integration that is only described in a general way and lacks distinctions between the different levels at which architectures are created.

The approaches we found so far put a strong focus on EA(M) or ASD, which is then extended by aspects of the other. However, the approaches lack specific details on how the interaction between EA(M) and ASD teams should work in practice.

5. Conclusions & Future Work

Despite the fact that the idea to integrate ASD and (enterprise) architecture is not entirely new, existing approaches usually focus on software development. Only a few approaches starting on the EA level could be found. None of them does, however, try to enable the agile creation of the EA while at the same time focusing on the collaboration with agile development teams. As the developed integration answers two different questions – whether and how agile methods, i.e. Scrum can be used to create architecture deliverables and how enterprise architects can collaborate with agile software development teams – it is thus unique and overcomes one of the main shortcomings of existing approaches. Instead of gradually extending Scrum by more and more aspects, two different procedure models, one to be used on the Enterprise Strategic and Segment Architecture Level and another for the Capability Architecture Level, are proposed. The integration does hence benefit from the broad range of topics covered by TOGAF, e.g. the portfolio management or the consideration of innovations requiring architecture change.

Besides its advantages, we are aware that our integration possesses certain limitations, which at the same time offers pivots for future work. As the integration is entirely based on expert interviews, it would be interesting to see how far the solution adheres to or can be backed up by ideas and concepts from existing approaches. Often the interviewees proposed distinctive possibilities to overcome a specific problem. Should any shortcomings of the solution be revealed when applying it in practice, these alternatives should be considered. In the end, it is this application of the integration (i.e. the Demonstration and Evaluation of the DS process) that has to prove, whether it fulfils the objectives stated in Subsection 3.2 and whether concepts from Scrum are useful for EA(M). First steps towards this integration have been taken at the participating automotive OEM, the results of which the authors look forward to present in the form of a case study. Future work should also focus on providing additional cases in comparison.

Acknowledgements

This paper has been written in cooperation with Detecon International GmbH, Germany. In particular we would like to thank Mr Philip Peters and Mr Frank Dräger for the continuous support and fruitful exchange.

References

[1] S. Buckl, F. Matthes, I. Monahov, S. Roth, C. Schulz, and C. M. Schweda, "Towards an agile design of the enterprise architecture management function," in *EDOCW*. IEEE Computer Society, 2011, pp. 322–329.

[2] H. P. Breivold, D. Sundmark, P. Wallin, and S. Larsson, "What does research say about agile and architecture?" in *ICSEA*. IEEE Computer Society, 2010, pp. 32–37.

[3] M. M. Lankhorst and H. A. Proper, "Agile architecture," in *Agile Service Development: Combining Adaptive Methods and Flexible Solutions*, M. M. Lankhorst, Ed. Springer, 2012, pp. 41–57.

[4] M. M. Lankhorst, W. P. M. Janssen, H. A. Proper, and M. W. A. Steen, "Introducing agile service development," in *Agile Service Development: Combining Adaptive Methods and Flexible Solutions*, M. M. Lankhorst, Ed. Springer, 2012, pp. 1–15.

[5] J. Sutherland and K. Schwaber, "The scrum guide," 2013 [cited 2014-06-11], https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf.

[6] "The Open Group Architecture Framework (TOGAF) Version 9.1," 2011 [cited 2013-12-03], https://www2.opengroup.org/ogsys/jsp/publications/PublicationDetails.jsp?catalogno=I112.

[7] D. West, T. Grant, M. Gerush, and D. D'Silva, *Agile Development: Mainstream Adoption Has Changed Agility*. Forrester Research, 2010.

[8] B. W. Boehm, "Get ready for agile methods, with care," *IEEE Computer*, vol. 35, no. 1, pp. 64–69, 2002.

[9] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *JMIS*, vol. 24, no. 3, pp. 45–77, 2008.

[10] U. Flick, *An Introduction to Qualitative Research*, 4th ed. SAGE Publications Ltd, 2009.

[11] S. Buckl, F. Matthes, and C. M. Schweda, "A technique for annotating ea information models with goals," in *EOMAS*, ser. LNBIP, vol. 63. Springer, 2010, pp. 113–127.

[12] M. Lankhorst, *Enterprise Architecture at Work: Modelling, Communication and Analysis*, 2nd ed. Springer, 2009.

[13] F. Ahlemann, E. Stettiner, M. Messerschmidt, and C. Legner, Eds., *Strategic Enterprise Architecture Management: Challenges, Best Practices, and Future Developments*. Springer, 2012.

[14] M. Op't Land, E. Proper, M. Waage, J. Cloo, and C. Steghuis, *Enterprise Architecture: Creating Value by Informed Governance*. Springer, 2009.

[15] S. Dyck and T. A. Majchrzak, "Identifying common characteristics in fundamental, integrated, and agile software development methodologies," in *HICSS*. IEEE Computer Society, 2012, pp. 5299–5308.

[16] K. Conboy, "Agility from first principles: Reconstructing the concept of agility in information systems development," *ISR*, vol. 20, no. 3, pp. 329–354, 2009.

[17] M. Cohn, *Succeeding with Agile: Software Development Using Scrum*. Addison Wesley, 2009.

[18] R. van Solingen, J. Sutherland, and D. de Waard, "Scrum in sales: How to improve account management and sales processes," in *AGILE*. IEEE Computer Society, 2011, pp. 284–288.

[19] K. Schwaber, "Waterfall, lean/kanban, and scrum," 2010 [cited 2014-06-11], http://kenschwaber.wordpress.com/2010/06/10/waterfall-leankanban-and-scrum-2/.

[20] D. J. Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press, 2010.

[21] A. Fleischmann, "What is s-bpm?" in *S-BPM ONE*, ser. CCIS, vol. 85. Springer, 2009, pp. 85–106.

[22] Z. Amiri, "Challenges and weaknesses of agile method in enterprise architecture," *IJCSES*, vol. 3, no. 6, pp. 37–45, 2012.

[23] E. Richardson, "What an agile architect can learn from a hurricane meteorologist," *IEEE Software*, vol. 28, no. 6, pp. 9–12, 2011.

[24] J. Watson, M. Rosen, and K. Guenther, "Are agile methods and enterprise architecture compatible? yes, with effort," *Cutter Consortium*, vol. 6, no. 11, pp. 1–25, 2005.

[25] M. Yaghoubi and M. Babanezhad, "Software developing with agile methods and combination of architecture," *IJCA*, vol. 65, no. 19, pp. 33–37, 2013.

[26] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change*, 2nd ed. Addison Wesley, 2004.

[27] J. Grundy, "Architecture vs agile: competition or co-operation?" in *Agile Software Architecture: Aligning Agile Processes and Software Architectures*, M. A. Babar, A. W. Brown, K. Koskimies, and I. Mistrik, Eds. Morgan Kaufmann, 2013, pp. XXI–XXVII.

[28] M. W. Lines and S. Ambler, *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise*. IBM Press, 2012.

[29] Z. Durdik, "Architectural modelling in agile methods," in *WCOP*, 2010, pp. 23–30.

[30] I. Hadar and S. Sherman, "Software architecture process in agile development methodologies," in *Information Systems (ILAIS) Conference*, 2012, pp. 78–85.

[31] S. Ambler, "Ibm agility@scale: Become as agile as you can be," 2009 [cited 2014-06-01], https://www14.software.ibm.com/iwm/web/cc/imc/rational/papers/Agility_at_scale.pdf.

[32] ——, "Agility@scale: Strategies for scaling agile software development," 2010 [cited 2013-12-12], https://www.ibm.com/developerworks/community/blogs/ambler/entry/agile_and_enterprise_architecture.

[33] C. Edwards, "Scrum based enterprise architecture planning process," 2007 [cited 2013-11-03], http://www.agileea.com/Whitepapers/2007-04-01-AEA_SCRUM_based_EA_Planning_Process.pdf.

[34] ——, "Agile enterprise architecture," 2006 [cited 2013-11-03], http://www.agileea.com/Whitepapers/2006-12-14-AgileEnterpriseArchitectureV1.00-Part1.pdf.

[35] E. Alali, "Agile software development under togaf," 2013 [cited 2013-11-03], http://goadingtheitgeek.blogspot.de/2013/02/agile-software-development-under-togaf.html.

[36] N. Ismail, "Togaf and scrum... where next," 2010 [cited 2013-11-03], http://naeemis.blogspot.de/2010/12/togaf-and-scrumwhere-next-prelimanary.html.

[37] Enterprise Architects Ltd, "Benefits led enterprise architecture method (bleam)," 2010 [cited 2013-11-03], http://www.enterprisearchitects.eu/ea/bleam.

[38] U. Bombosch, "Agilitaet trotz komplexer architekturen," 2012 [cited 2013-11-03], http://jaxenter.de/artikel/Agilitaet-trotz-komplexer-Architekturen.